A Two-level Particle Swarm Optimization for Traveling Salesman Problem with Neighborhood

Yihjia Tsai
Department of Computer Science and Information Engineering
Tamkang University
151 Injuang Road, Tamsui, New Taipei
Taiwan
iaicLab.tku@gmail.com

ABSTRACT

In this paper, a two-level particle swarm optimization approach is proposed to solve the problem of traveling salesman problem with neighborhood (TSPN). A direct application of the algorithm is in a scheduling mobile robot gathering data from a set of spatially distributed wireless sensors in the euclidean plane. The proposed two-level approach is a combination of both discrtet and continuous optimization schemes. The first level is a discrete PSO which determines the visiting order of fixed sensors while the second level of PSO solves the exact visiting point within communication range for each sensor. Two heuristic operations are used to facilitate the convergence of the proposed algorithm. The first is to rearrange the visiting order when an intermediate node is within communication range along the visiting path. The second operation is to swap two crossing visiting paths to reduce the overall path length. Simulation results have shown that the proposed algorithm is able to converge to the optimal solution for a intermediate size networks.

Categories and Subject Descriptors

I.2.8 [**ARTIFICIAL INTELLIGENCE**]: Problem Solving, Control Methods, and Search – *heuristic methods, scheduling.*

General Terms

Algorithms.

Keywords

Particle Swarm Optimization, Traveling Salesman Problem with Neighborhood, Wireless Sensor Networks.

1. INTRODUCTION

Particle swarm optimization (PSO) method [9] has been used in solving optimization problems in wireless sensor networks (WSNs), for example, sensor deployment [10], cluster head selection [7], coverage [2], [17], [19], and localization [4]. The problem of scheduling mobile elements for data gathering can be regarded as a variant of the Traveling Salesperson Problem with Neighborhoods (TSPN) [1] and is also known to be NP-hard [13]. Results of using genetic algorithm to determine the visiting point the neighborhood of a given node from a pre-determined visiting order is given in [22]. The optimization problem does not involve the permutation problem of determining the node visiting order. In this paper, the problem of both determining the sensor visiting order and the visiting point position within the neighborhood of visited node is studied and a two-level PSO algorithm is proposed.

An example application of the proposed algorithm is for scheduling data gathering robots in WSNs. One of the challenging issues in the design and operation of WSNs is system lifetime elongation. In order to achieve it, extensive researches studied the use of mobile elements to gather data from deployed static sensing nodes. These mobile elements are called data mules in some studies. There can be multiple data mules moving in the sensor network for data gathering. The mobility pattern of mobile sink can be either random or predetermined or autonomous. A recent review [21] has surveyed the mechanisms of node mobility to improve system lifetime. Some theoretical analysis on the use mobile entities called MULEs to pick up data from sensing nodes and deliver to data sinks is given in [15]. The analysis is based on random walk mobility model and explores the relationship between mules and sensor memory capacities. This finding indicates that for a constant data delivery success rate, increasing the memory capacity in MULEs is more efficient than adding memory to sensor nodes.

In [12], the authors studied a system with multiple sinks moving along predetermined paths to a new location with higher sensor energies for system lifetime improvements. Instead of moving to within communication range of every sensor node, mobile elements can gather data from a region of sensors through multi-hop routing mechanism. For example, in [5], an integer linear program (ILP) is used to determine new locations of the mobile elements and a flow-based routing (FBR) protocol is used to calculate multi-hop routing from sensor nodes near mobile elements. ILP is formulated to minimize the maximum energy spent by a sensor node in a round. The solution of ILP indicates the mobile element's next location. At the beginning of each round new mobile elements' positions are computed and they remain fixed during that round, while the mobile elements gather data from the network. Another approach is to first gather data from certain number of sensors within a region to a representative

GECCO 2011, July 12–16, 2011, Dublin, Ireland.

sensor called cluster head, and cluster head is responsible for relaying data to the mobile element when in communication range [14]. This approach involves several issues: the selection of cluster head, the local routing path from sensor nodes to cluster head and the routing path of mobile element to gather data from cluster heads. In [11], the issues of cluster head selection and local routing path calculation are formulated as a maximum-flow problem and the visiting path of mobile element is determined by a divide-and-conquer process and the resulting path leads from a region of fixed sensors to another in a straight line fashion. All nodes within communication range of the straight visiting path act as cluster heads. The issue of determine the visiting path is modeled as a minimum-path data gathering (MPDG) problem in [20] and a heuristic algorithm based on convex hull construction is used to calculate the visiting path. Planning a shortest visiting path that passes all pre-defined nodes in unknown order is known as Traveling Salesman Problem (TSP) [6], and this NP-hard problem has received extensive studies over the years.

The rest of this paper is organized as follows: System model is introduced in Section 2 and Section 3 briefly outlined PSO and its formulation. The proposed PSO model is presented in Section 4 and simulation results of the proposed model are discussed in Section 5. Finally concluding remarks and future applications in WSNs are given in Section 6.

2. SYSTEM MODEL

A group of fixed sensors with sensing radius *Rs* and communication radius *Rc = 2Rs* are deployed forming a network *G={V,E}*, where *V* is the set of sensor nodes and *E* is the set of communication links. The location of sensor *i*, denoted by (x_i, y_i), is known in advance. The Euclidean distance between any two sensor nodes *i* and *j* is denoted by *d(i,j)*. Assuming that some cluster head selection algorithm is applied and a set of head sensors is responsible for collecting data from nearby nodes. Cluster head selection algorithm can be the well-known LEACH [8] or its variants. A mobile element endowed with more computing and storage capabilities and has a communicating radius *Rc* is responsible for collecting data form those cluster heads. These cluster head are spatially separated and their locations are known to the mobile element. The task of the mobile element is to schedule an efficient (shortest) data gathering route visiting all cluster head nodes. In its generic form, this mobile data gatherer scheduling problem (MDGSP) can be regarded as a special case of TSPN.

3. BRIEF OVERIEW OF PSO

PSO algorithm models the flocking behavior of a swarm of birds by k artificial particles. Each particle is a candidate solution to the unknown optimization problem moving in an h-dimensional hyperspace. Particle *i* is represented by an *h*-dimensional position vector $\mathbf{x}_i = (x_1^i, x_2^i, \ldots, x_h^i)^\top$ and a velocity vector $\mathbf{v}_i = (v_1^i, v_2^i, \ldots, v_h^i)^\top$. An object function $f(\mathbf{x})$, where $f : \mathbf{R}^h \to \mathbf{R}$, evaluates each particle's fitness value (closer) to the global solution of the optimization problem. An *h*-dimensional vector variable *gbest* stores the position of the best particle while particle *i*'s historical best position is stored in vector variable $pbest^i$. At iteration *t*, the position and the velocity vectors are updated using (1) and (2). The update process is repeated until either an acceptable *gbest* is achieved or a fixed number of iterations $t_{\max}$ is reached.

$$\mathbf{v}_i(t+1) = w_1 \cdot \mathbf{v}_i(t) + w_2 \cdot r_1(t) \cdot \left(pbest_i - \mathbf{x}_i(t)\right) + w_3 \cdot r_2(t) \cdot \left(gbest - \mathbf{x}_i(t)\right) \quad (1)$$

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1), \quad (2)$$

where *i* = 1, 2, ... is the index of particle. Constants w_1, w_2 and w_3 are predetermined, and $r_1(t)$ and $r_2(t)$ are uniformly distributed random numbers between [0,1]. when applying this classical PSO formulation in solving the problem of scheduling mobile elements, the position vector becomes discrete and represents the visiting order of fixed sensing nodes.

4. PROPOSED PSO FORMULATION

As outlined in [3], the use of discrete PSO algorithm for solving TSP requires several moajor modifications. Instead of a continuous high dimensional space vector, the position vector now represents the node visiting order and a velocity vector is a transposition which will yield another position vector when applied to a position vector. The key issue in facilitating the convergence of the algorithm lies in the combination of local and global best positions and the current position vector in giving the next position vector. Furthermore, as inspired by the use of swap operation in PSO to solve the TSP problem [18], authors in [16] suggested the use of a swarm particle to perform swap operation and communicate these results to other particles through the use of global best variable. This approach, however, requires a large number of iterations before a feasible visiting order emerges from the swap operation when the number of nodes is large.

In order to solve the mobile data gatherer scheduling problem, which is modeled as a TSPN problem, we proposed the use of a two-level PSO approach. The first level of PSO determines the visiting order of sensors while the second level determines the visiting location within the communication range of a visited node. In determining the visiting order, a position vector is a permuted list of fixed sensor ids and the difference between two permutations is represented by a velocity vector.

Particle *i*'s position vector represents the visiting order of *n* fixed sensors and is denoted by an *n*-dimensional position vector $\mathbf{x}_i = (x_1^i, x_2^i, \ldots, x_n^i)^\top$. This vector indicates the traveling cycle of $x_1^i \to x_2^i \to x_3^i \ldots \to x_n^i$. For each traveling order particle, a second group of particles is used to solve for the hitting point problem. A hitting point particle is represented by an *n*-dimensional vector $\mathbf{y}_j = (\alpha_1^i, \alpha_2^i, \ldots, \alpha_n^i)^\top$ where α_k representing the angle of hitting point for sensor *k*. As shown in Figure 1, the starting position of mobile element is represented by node *s*, when visiting node *i*, the position of hitting point in the circle of communication range around node *i* can be represented by its angle and is denoted by α_i.

This second level of PSO solves the optimization problem with continuous variables for a given visiting order. However, the solution of the second level of PSO may change the order of visits. For instance, a path connecting node i and $(i+1)$ may passing through the proximity (communication range) of node j, therefore, the order of visit should be changed from $i \rightarrow (i+1)$ to $i \rightarrow j \rightarrow (i+1)$, as illustrated in Figure 2.

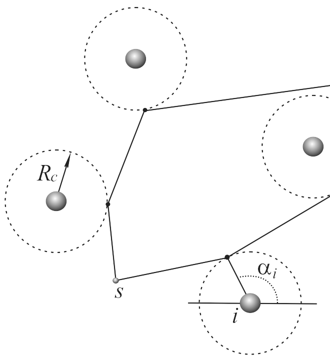


Figure 1. A mobile element starting from point *s* plans a traveling cycle visiting 5 sensor nodes.

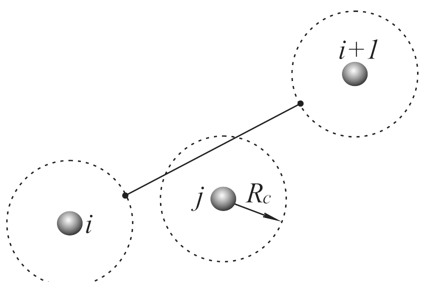
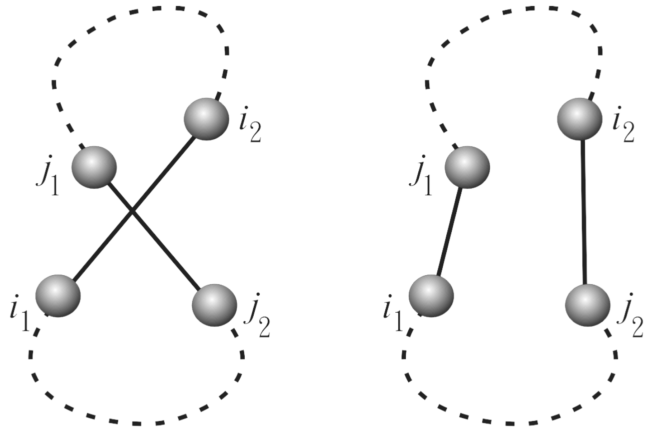


Figure 2. A visiting path from node *i* to node *i+1* crosses the communication range of node *j*. The order of visits is changed from $i \rightarrow (i+1)$ to $i \rightarrow j \rightarrow (i+1)$.

Inspired by the study of using swap operation to find the solution of TSP [11], another refinement of the visiting order is done by exchange the visiting order when a crossing is found along the path of visits. As illustrated in Figure 3(a), when a crossing on two path segments connecting nodes (i_1, i_2) and (j_1, j_2) is found, a swap operation that re-connects node i_1 to j_1 and i_2 to j_2, as indicated in Figure 3(b), will ensure a shorter overall path length according to triangle inequality. As suggested by [16], this operation can be time consuming for any randomly permuted visiting order and therefore is not to be used for every swarm particles.

In the proposed two level PSO approach, as indicated in Table 1, instead of performing both proximity and swap operations within swarm particles, we refine the global best solution by those two operations and use that as the next position vector for a visiting



(a) a crossing point between path segments (i_1,i_2) and (j_1,j_2).

(b) after swap operation, path segments now connect i_1 to j_1 and i_2 to j_2.

Figure 3. Swap operation.

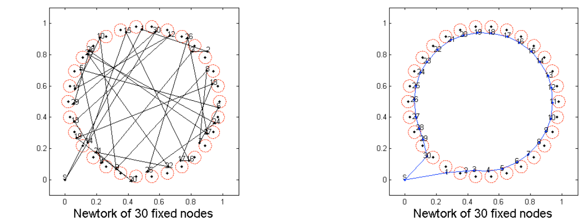
order swarm particle. While performing the swap operation, we impose a limit in the overall number of crossings removed and some rearrangement is needed in keeping the path connected.

Table 1 The proposed PSO algorithm

1	Initialize positions of both mobile element and sensors
2	Initialize particles with random visiting orders
3	Repeat until convergence or a max count is arrived
4	For each visiting order particle
5	Optimize hitting positions
6	Calculate local best visiting order
7	Calculate global best visiting order
8	Combine to form next visiting order
9	Perform swap operations on global best visiting order and assign the outcome to one of the combined results
10	Perform proximity operation on all newly formed visiting orders
11	Assigning those outcomes to visiting order particles
12	End repeat

5. SIMULATION RESULTS

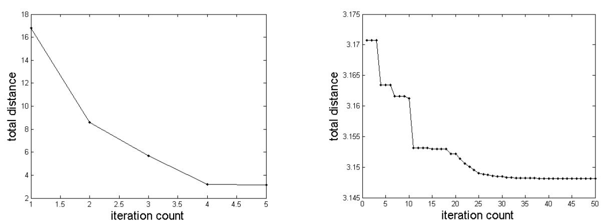
Simulations are performed by using Matlab2009a. Two test cases are created to study the effectiveness of the proposed algorithm in finding solution. The first test case consists of a set of set of 30 fixed sensors distributed uniformly in a circular fashion around (0.5, 0.5) with the same communication range of 0.04, and the initial visiting order is randomly arranged, as shown in Figure 4(a). The initial position of the mobile element is located in coordinate (0,0), which is outside the ring formed by those sensors. This configuration tests if the proposed algorithm can find the optimal visiting path along the inner boundaries formed by those disks. Figure 4(b) illustrates the resulting visiting path of the proposed algorithm, which traces the inner boundaries formed by all the communication circles.



(a) Initial random visiting order. **(b) Resulting visiting path.**

Figure 4. A test case consists of 30 uniformly distributed sensors around a circle.

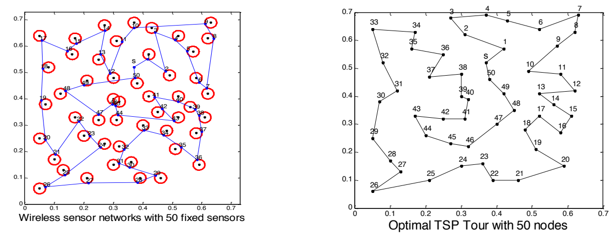
In Figure 5(a) the total distance of the visiting path is plotted against the number of iterations for the test case. The first level of PSO uses 10 particles and 5 iterations while the second level of PSO uses 8 particles and 50 iterations. Shown in Figure 5(b) is the convergence of the second level hitting location optimization for the best visiting order. The limiting number of iterations in the swap operation is 1500.



(a) convergence of total distance **(b) convergence of hitting location PSO**

Figure 5. Convergence of the total distance for the test case, the converged distance is 3.1482

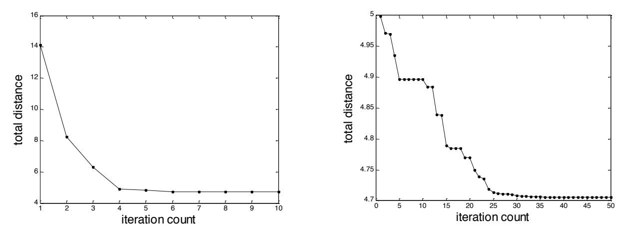
The second case was aimed at investing the potential of the proposed PSO algorithm in solving benchmark examples from the well known TSPLIB [23]. Optimal solutions for some examples are known. We pick the 51 nodes dataset eil51 and use the first node as the mobile element's location while the remaining 50 nodes denoting sensor nodes. In the benchmark data, coordinates are integers between 0 and 80, therefore we divide them by 100 and a communication range of 0.02 is chosen for all sensor nodes. The optimal tour distance for dataset eil51 is 426. The resulting tour calculated from the proposed PSO algorithm is illustrated in Figure 6(a) while in Figure 6(b) is the optimal tour. The convergence of total distance is plotted against the number of iterations, as shown in Figure 7(a). For the best visiting order, the convergence of the total distance in the second level hitting position optimization is illustrated in Figure 7(b). These convergence results are similar to the first test case for 20 nodes. The parameter for the second level of PSO are the same, 8 particles and 50 iterations, however, the first level of PSO uses 10 particles and instead of 5, this test case uses 10 iterations and the limiting number of iterations in the swap operation is 5000.



(a) Visiting path calculated by the proposed algorithm, total distance is 4.70 **(b) Optimal visiting path, total distance is 4.62**

Figure 6. Result of using benchmark dataset eil51

The difference in the limiting number of swap operations indicates that in order to find the best visiting order, a large number of swap operations is needed to escape from the local minimum as the problem size grows from 30 to 50.



(a) convergence of total distance **(b) convergence of hitting location PSO**

Figure 7 Convergence of the total distance for the test case, the converged distance is 4.70

6. CONCLUSIONS

In this paper, a two level hybrid PSO approach for solving the problem of traveling salesman with neighborhood is proposed. The problem is a special case of TSP and has been addressed in [22] however, their approach is based on a predetermined order of visits. The issue of discrete optimization problem in determining the optimal visiting order is not deal with. Permutation problem is known to be harder to converge as compared with continuous variable problem for evolutionary-based algorithms. From simulation results we know that the proposed hybrid PSO algorithm combines proximity and swap operations can facilitate the convergence of PSO in solving this mobile data gatherer scheduling problem.

7. REFERENCES

[1] Arkin, E. M., and Hassin, R. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55, 3 (Dec. 1994), 197-218.

[2] Ab. Aziz, N. A., Mohemmed, A. W., and Zhang, M. Particle Swarm Optimization for Coverage Maximization and Energy Conservation in Wireless Sensor Networks. *Applications of Evolutionary Computation, Lecture Notes in Computer Science*. 6025/2010, 2010, 51-60.

[3] Clerc, M. Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem. *New Optimization Techniques in Engineering*. Springer, Heidelberg, Germany:, 2004, 219-239.

[4] Chuang, P.-J., and Wu, C.-P. An Effective PSO-Based Node Localization Scheme for Wireless Sensor Networks. In the Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies (Otago, Japan). 2008, 187 – 194.

[5] Gandham, S. R., Dawande, M., Prakash R., et al. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In IEEE GLOBECOM '03, 2003, 377-381.

[6] Greco, F., Ed. Traveling Salesman Problem. In-Tech, Vienna, Austria, 2008.

[7] Guru, S. M., Halgamuge, S. K., and Fernando, S. Particle Swarm Optimisers for Cluster formation in Wireless Sensor Networks. In Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP). 2005, 319-324.

[8] Heinzelman, W., Chandrakasan, A., and Balakrishnan, H. Energy-efficient Communication Protocol for Wireless Microsensor Networks. In Proceedings of 33rd Hawaii International Conference on System Sciences, 2000.

[9] Kennedy, J., and Eberhart, R. C. Particle swarm optimization. In Proceedings of IEEE International Conference on Neural Networks (Perth, WA, Australia). 1995, 1942-1948.

[10] Kukunuru, N., Thella, B. R., and Davuluri, R. L. Sensor Deployment Using Partice Swarm Optimization. *International Journal of Engineering Science and Technology*, 2, 10, 2010, 5395-5401.

[11] Ma, M., and Yang, Y. SenCar: An Energy-Efficient Data Gathering Mechanism for Large-Scale Multihop Sensor Networks. *IEEE Transactions on Parallel and Distributed Systems*, 18, 10, 2007, 1476-1488.

[12] Marta, M., and Cardei, M. Improved sensor network lifetime with multiple mobile sinks. *Elsevier Journal of Pervasive and Mobile Computing*, 5, 5, 2009, 542–555.

[13] Papadimitriou, C. H., The Euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, 4, 3, 1977, 237-244.

[14] Saad, E. M., Awadalla, M. H., and Darwish, R. R. Adaptive Energy-Aware Gathering Strategy for Wireless Sensor Networks. *International Journal of Computers*, 2, 2, 2008, 148-157.

[15] Shah, R. C., Roy, S., Jain, S. et al. Data MULEs: modeling a three-tier architecture for sparse sensor networks. In IEEE Workshop on Sensor Network Protocols and Applications (Anchorage, Alaska, USA). 2003, 30-41.

[16] Shi, X. H., Liang, Y. C., Lee, H. P. et al. Particle swarm optimization-based algorithms for TSP and generalized TSP. *Information Processing Letters*. 103, 5, 2007, 169-176.

[17] Tian, W., and Liu, J. A Novel Optimization Method for the Maximum Coverage Sets of WSN. In International Conference on Wireless Networks and Information Systems (Shanghai, China). 2009, 125-128.

[18] Wang, K.-P., Huang, L., Zhou, C.-G. et al. Particle Swarm Optimization for Traveling Salesman Problem. In Proceedings of the Second International Conference on Machine Learning and Cybernetics (Xi'an, China). 2003, 1583-1585.

[19] Wang, X., Ma, J.-J., Wang, S. et al. Distributed Particle Swarm Optimization and Simulated Annealing for Energy-efficient Coverage in Wireless Sensor Networks. *Sensors*, 7, 5,2007, 628-648.

[20] Wu, F.-J., Huang, C.-F., and Tseng, Y.-C. Data Gathering by Mobile Mules in a Spatially Separated Wireless Sensor Network. In Proceedings of the Tenth International Conference on Mobile Data Management: Systems, Services and Middleware (Taipei, Taiwan). 2009, 293-298.

[21] Yang, Y., Fonoage, M.I., and Cardei, M. Improving network lifetime with mobile wireless sensor networks. *Computer Communications*. 33, 4, 2009, 409-419.

[22] Yuan, B., Orlowska, M., and Sadiq, S. On the Optimal Robot Routing Problem in Wireless Sensor Networks. *IEEE Transactions on Knowledge and Data Engineering*. 19, 9, 2007, 1252-1261.

[23] TSPLIB, www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/, 2010.